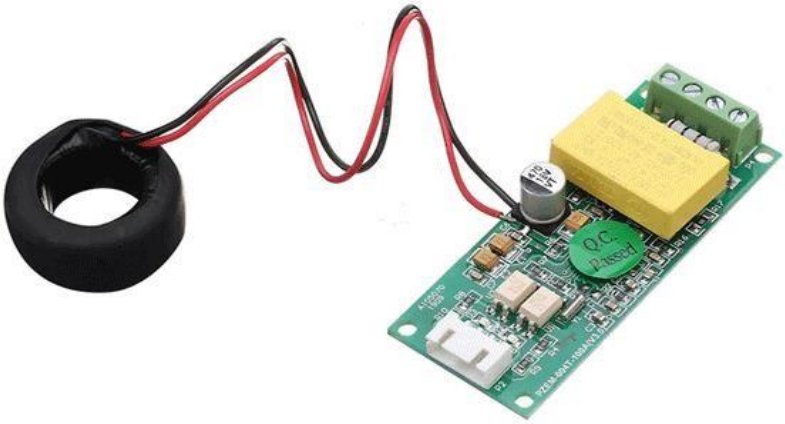
Buku Panduan

SMART PJU



ABIZAR AZHAR SHALEH

ANISHA FITRI PERTIWI

RIFALDI ILHAN MANSIZ

PPTIK - ITB

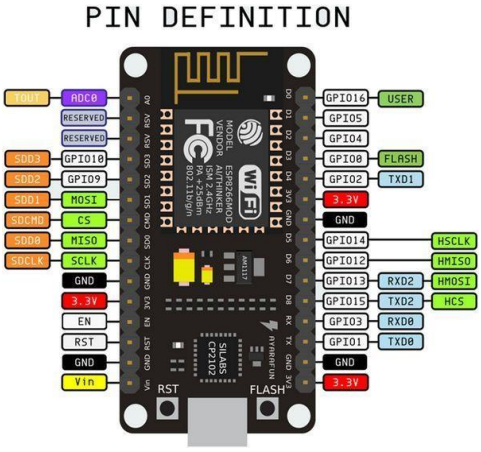
BAB I

PENDAHULUAN

Penerang Jalan Umum (PJU) atau biasa disebut dengan lampu jalan merupakan fasilitas umum yang berfungsi sebagai penerangan jalan dimalam hari sehingga para pejalan kaki, pesepeda, dan pengguna kendaraan lainnya dapat melihat dengan lebih jelas jalan/medan yang akan dilalui pada malam hari, sehingga meningkatkan keselamatan lalu lintas dan keaamanan para pengguna jalan dari kegiatan/aksi kriminal.

1.1 NodeMCU ESP8266

NodeMCU adalah sebuah platform *Internet of Things* (IoT) yang bersifat *opensource*. Terdiri dari perangkat keras berupa *System On Chip* ESP8266 dari ESP8266 buatan **Espressif System**, juga *firmware* yang digunakan, yang menggunakan bahasa pemrograman *scripting* Lua. Istilah NodeMCU secara *default* sebenarnya mengacu pada firmware yang digunakan daripada perangkat keras development kit. NodeMCU telah me-*package* ESP8266 ke dalam sebuah *board* yang kompak dengan berbagai fitur layaknya mikrokontroler + kapabilitas akses terhadap Wifi juga chip komunikasi USB to serial. Sehingga untuk memprogramnya hanya diperlukan ekstensi kabel data USB persis yang **digunakan sebagai kabel data dan kabel *charging* smartphone Android.**



Gambar 1. 1 Pin NodeMCU ESP266

1.2 PZEM-004T

PZEM-004T adalah sebuah modul elektronik yang berfungsi untuk mengukur : Voltage / Tegangan, Arus, Daya, Frekuensi, Energi dan Power Faktor. Dengan kelengkapan fungsi / feature ini, maka modul PZEM-004T sangat ideal untuk digunakan sebagai project maupun eksperimen alat pengukur daya pada sebuah jaringan listrik seperti rumah atau gedung.

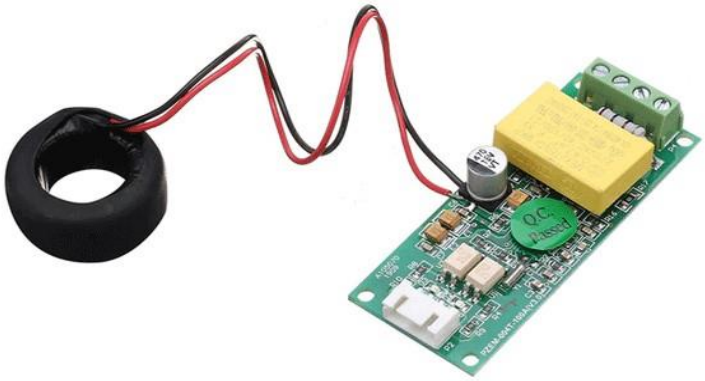
Modul PZEM-004T diproduksi oleh sebuah perusahaan bernama Peacefair, ada yang model 10 Ampere dan 100 Ampere. Harap berhati-hati karena wiring antara yang model 10 Ampere dengan 100 Ampere berbeda, jika salah bisa terjadi konslet atau hubungan arus pendek pada jaringan listrik.

a. Fungsi

- Fungsi pengukuran (voltage / tegangan, current / arus, active power).
- Power button clear / reset Energy (PZEM-004T V2.0)
- Power-down data storage function (cumulative power down before saving)
- Komunikasi Serial TTL
- Pengukuran Power / Daya : 0 ~ 9999kW
- Pengukuran Voltage / Tegangan : 80 ~ 260VAC
- Pengukuran Current / Arus : 0 ~ 100A

b. Spesifikasi

- Working voltage: 80 ~ 260VAC
- Rated power: 100A / 22000W
- Working Frequency: 45-65Hz
- Measurement accuracy: 1.0



Gambar 1. 2 PZEM-004T

1.1 Perbedaan antara PZEM-004T V2.0 dan PZEM-004T V3.0

- Pada V2.0 ada tombol untuk Reset Energi
- Pada V3.0 fungsi untuk Reset Energi menggunakan software, karenanya sudah tidak ada tombol push button untuk Reset Energi
- V3.0 merupakan upgrade versi V2.0 sehingga tingkat akurasi lebih baik ▪ Waktu konversi / pembacaan pada V3.0 lebih cepat dibandingkan V2.0.
- Protokol yang digunakan untuk komunikasi data sudah berbeda antara keduanya.

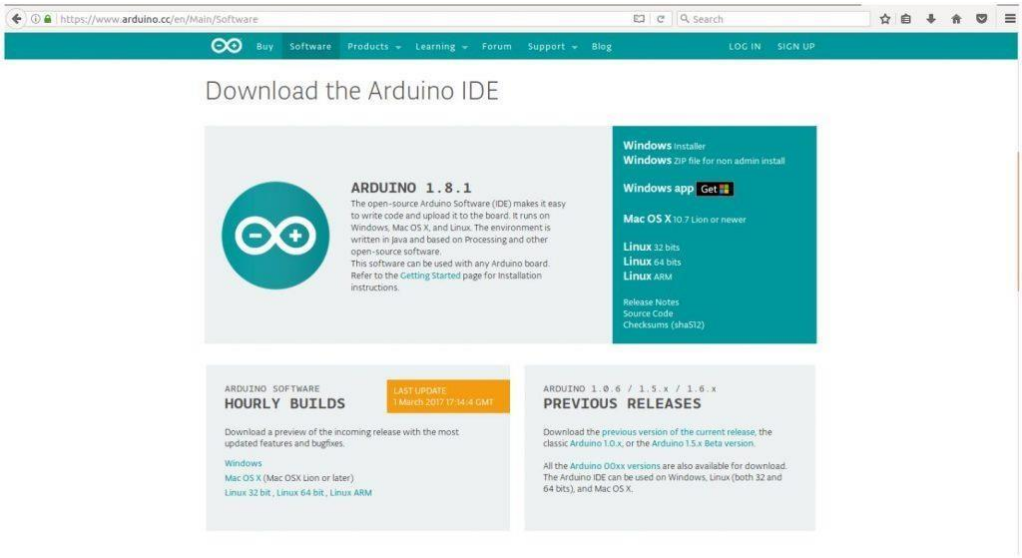
BAB II

Software Installation

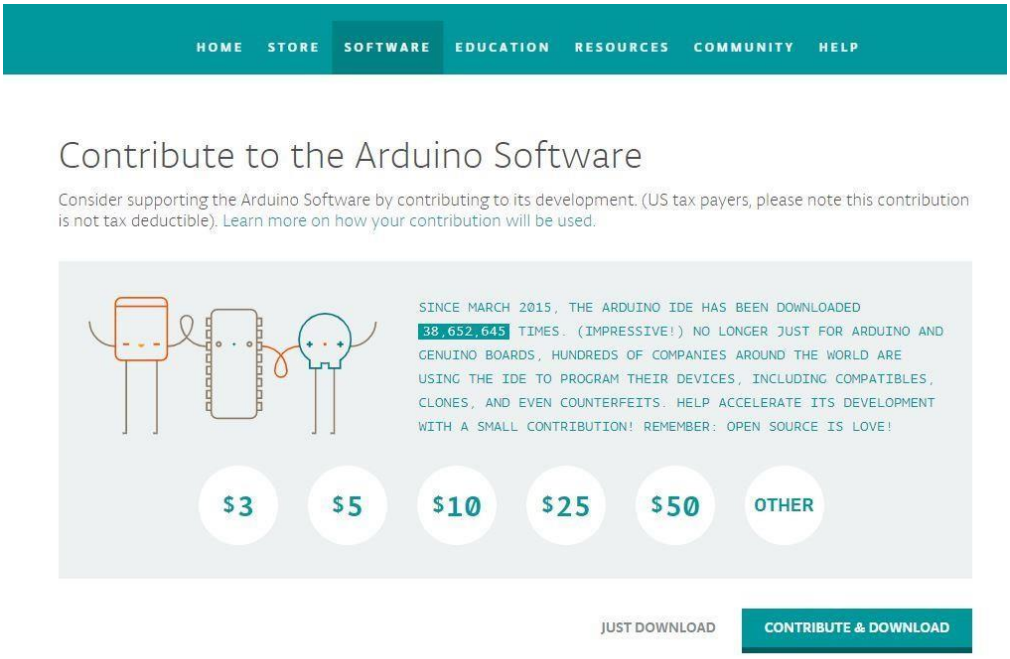
2.1 Download File Software Arduino IDE untuk Windows

Download Software Arduino IDE dengan mengklik link dibawah : https://www.arduino.cc/en/Main/Software.

Kemudian pilih ***Windows Installer*** paling atas. kemudian akan tampil pilihan Download. Silahkan klik ***Just Download*** atau ***Contribute dan Download***.



Gambar 2. 1 Klik *Windows Installer*



Gambar 2. 2 Klik Just Download atau Contribute & Download

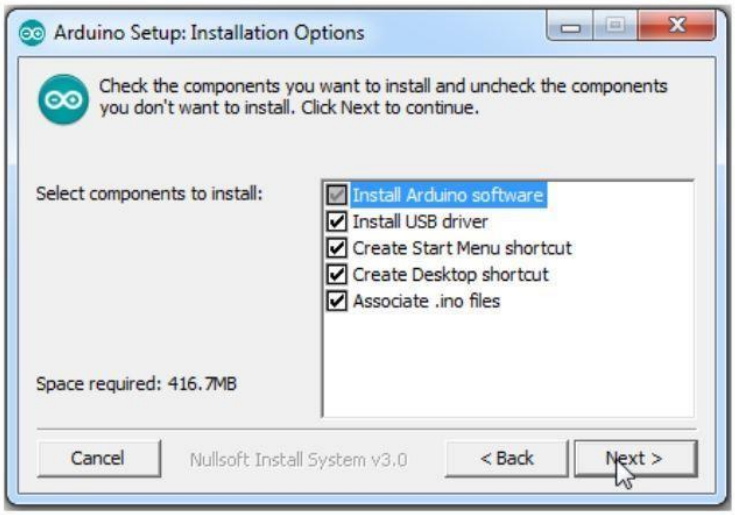
2.2 Persetujuan Instalasi Software IDE Arduino

Setelah selesai download silahkan buka dengan klik kiri dua kali atau klik kanan open kemudian akan muncul **License *Agreement atau Persetujuan Instalasi***, klik tombol ***I Agree*** untuk memulai install software Arduino IDE.

Gambar 2. 3 Klik *I Agree*

3 Pilihan Opsi Instalasi

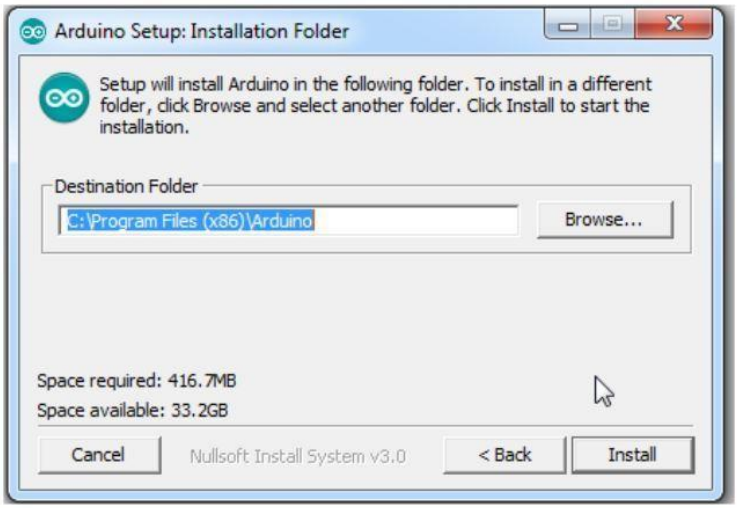
Untuk ***Installation Option*** pilih semua option dan klik tombol ***Next.***



Gambar 2. 4 Pilih Semua Option dan klik tombol *Next*

4 Pilih Folder

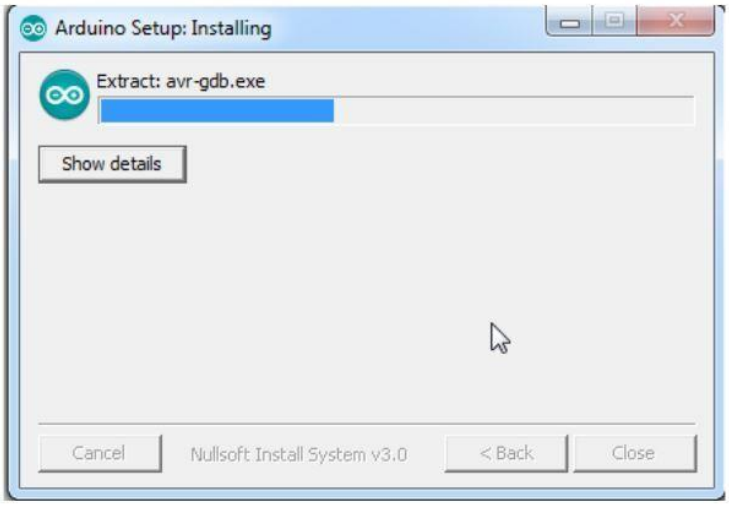
Installation Folder atau Pilihan Folder untuk memilih folder tempat menyimpan program arduino dan klik tombol **install** untuk memulai proses instalasi software.



Gambar 2. 5 Pilihan *Installation Folder* atau Pilihan Folder

5 Proses Extract dan Instalasi di mulai

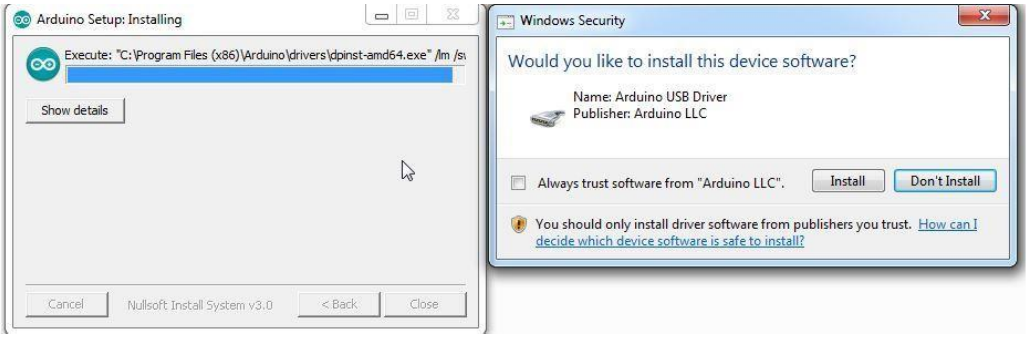
Proses instalasi di mulai,program di extract ke Windows.



Gambar 2. 6 Proses *Extract* dan Instalasi

6 Install USB Driver

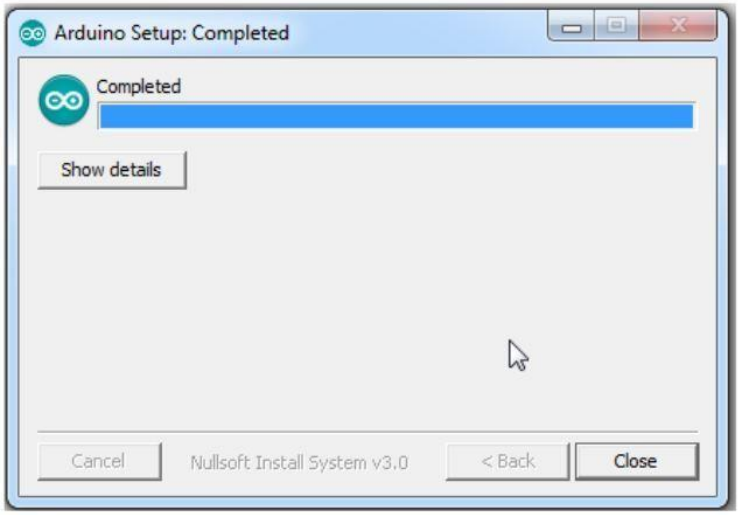
Saat proses instalasi sedang berlangsung akan muncul pilihan untuk install driver, pilih tombol instal,proses ini untuk mengenali dan melakukan komunikasi dengan board arduino melalui port USB.



Gambar 2. 7 *Install* USB Driver

7 Proses Instalasi Selesai

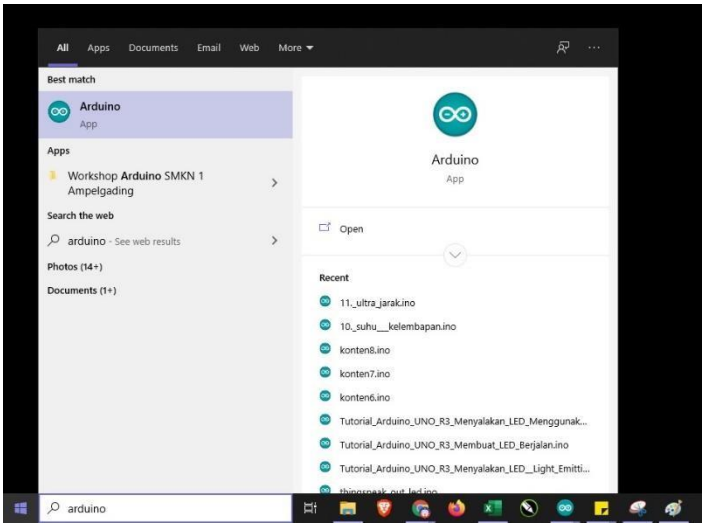
Setelah selesai silahkan klik close.



Gambar 2. 8 Proses Installasi

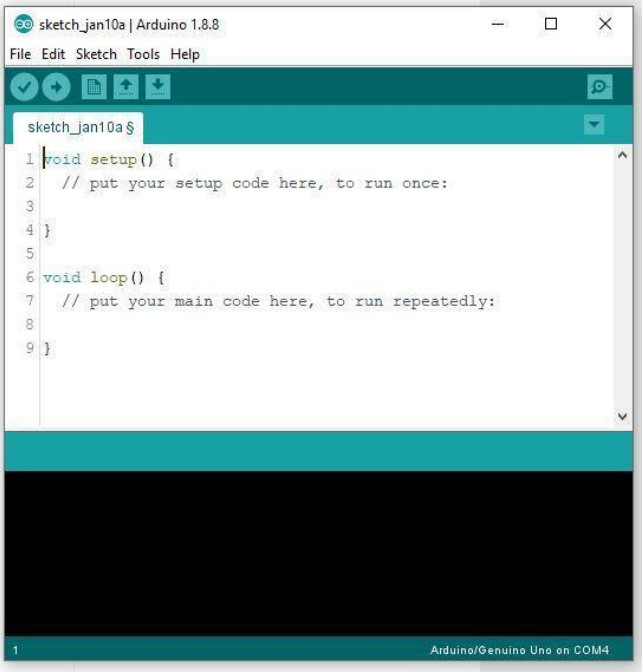
8 Start Software IDE Arduino

Proses instalasi Software Arduino sudah selesai terinstal di windows.Cek di Desktop Windows atau Start Menu untuk menjalankan Software IDE Arduino.Double Klik icon Arduino di desktop atau klik di Start Menu.



Gambar 2. 9 *Start Software* IDE Arduino IDE

9 Jendela Awal Software IDE Arduino



Gambar 2. 10 Tampilan *Software* Arduino IDE

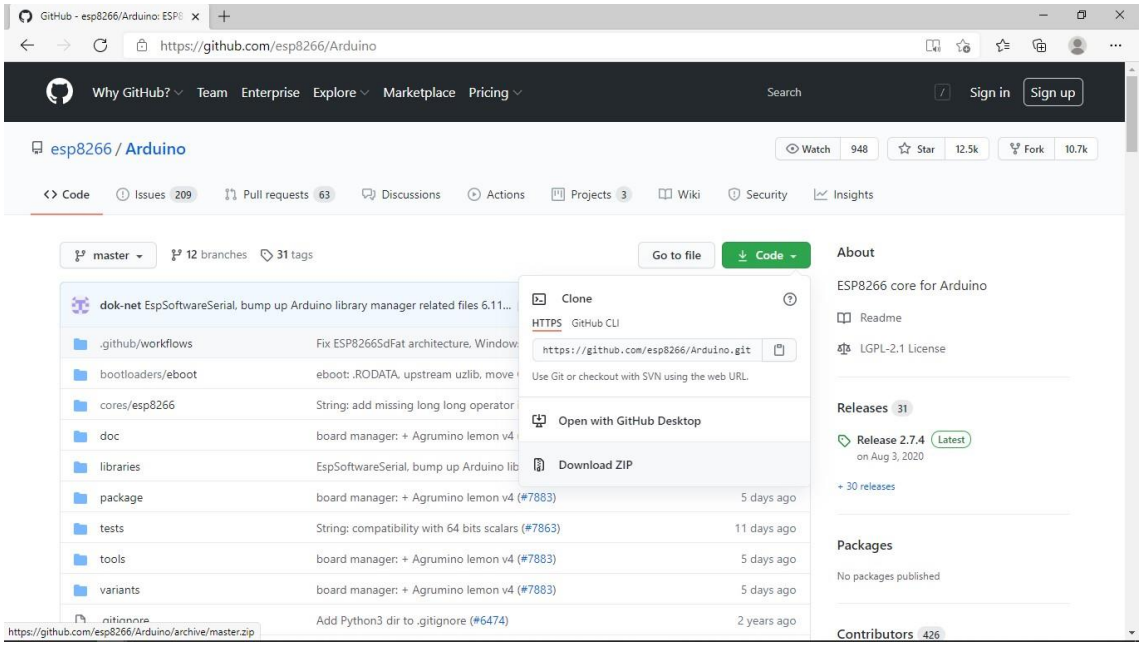
BAB III

LIBRARY

3.1 Download File Library ESP8266

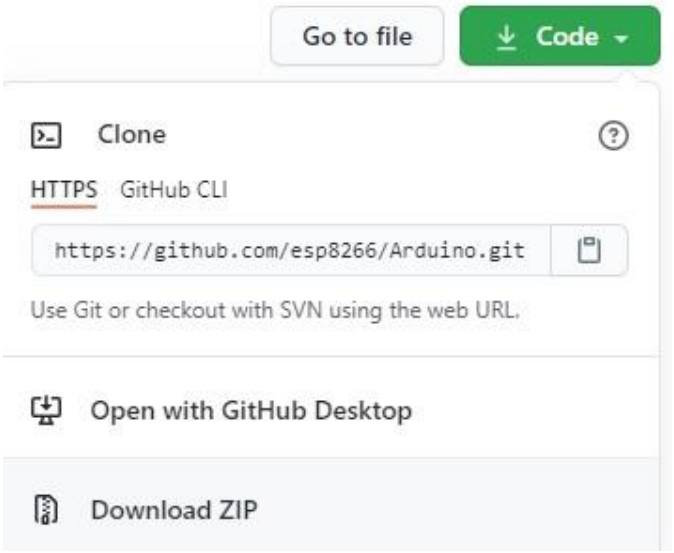
a. Download library ESP8266 dengan mengklik link di bawah :

GitHub - esp8266/Arduino: ESP8266 core for Arduino



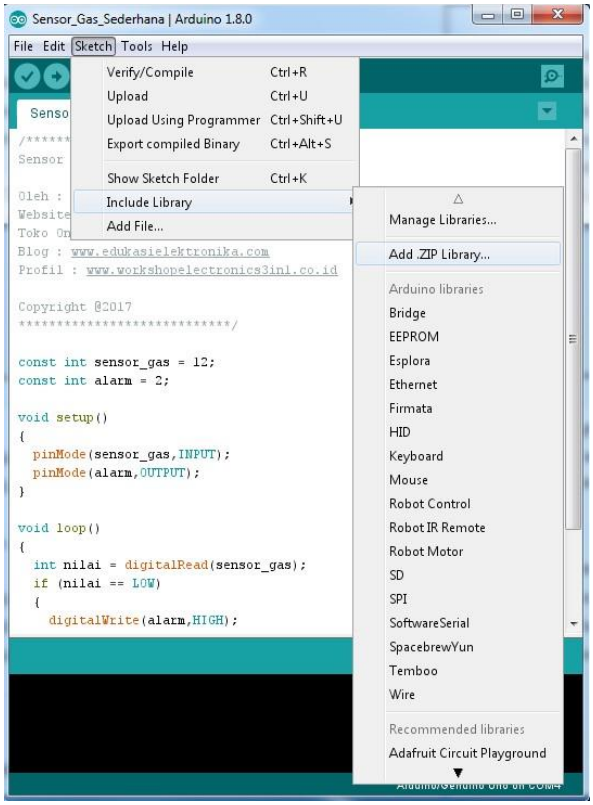
Gambar 3. 1 *Download File Library* ESP8266

Lalu klik “Download ZIP”



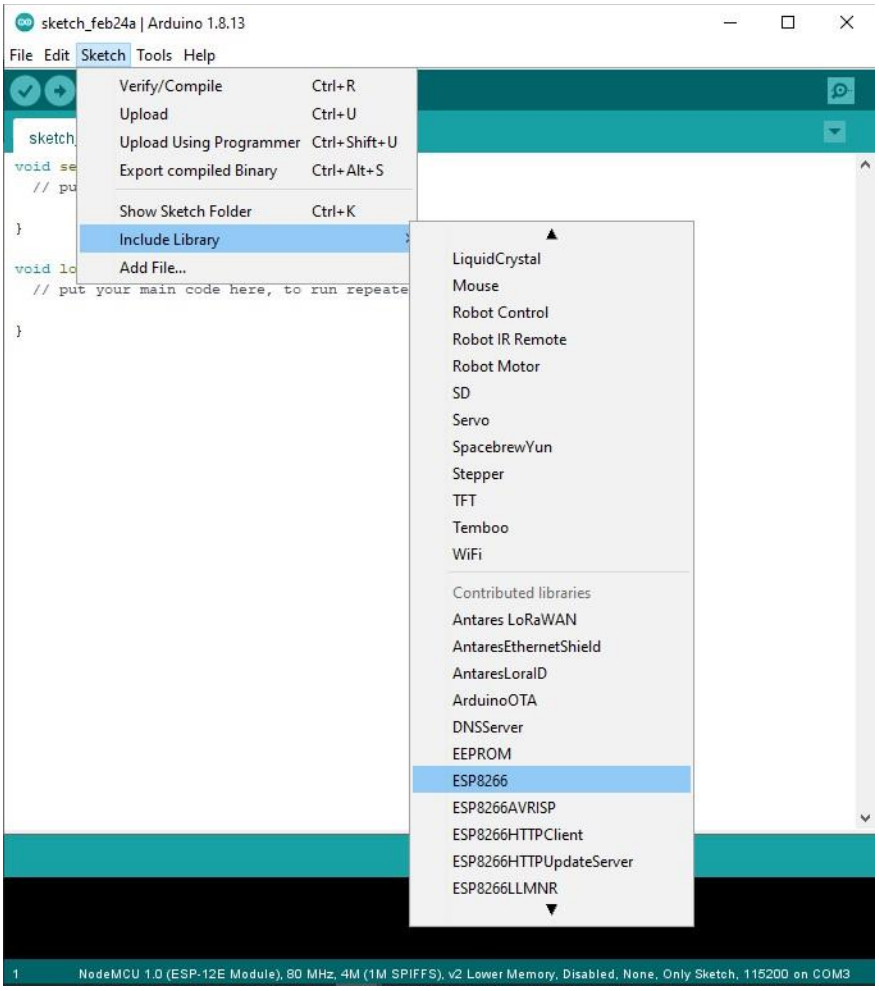
Gambar 3. 2 Klik *Download* ZIP

b. Buka Software atau Aplikasi **Arduino IDE.** Perhatikan gambar dibawah ini. Klik **add.ZIP** Library.



Gambar 3. 3 Klik add.ZIP *Library*

f. Lalu pilih File Library **ESP8266** yang sudah anda download tadi, Jika Berhasil akan muncul seperti gambar dibawah ini.

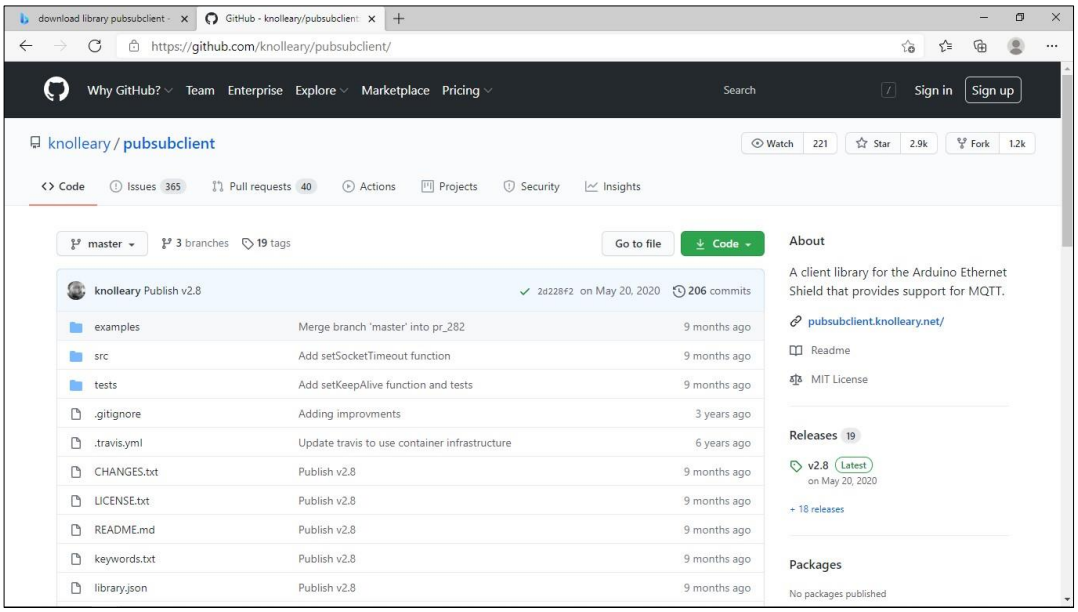


Gambar 3. 4 Tampilan Library berhasil terpasang

g. Library siap digunakan.

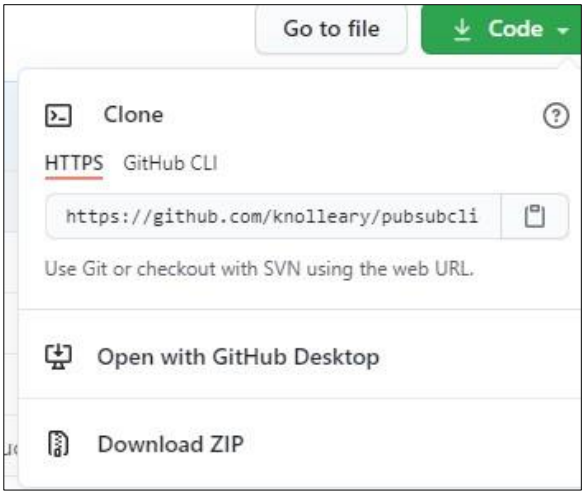
3.2 **Download File Library PubSubClient**

a. Download library PubSubClient dengan mengklik link di bawah : GitHub - knolleary/pubsubclient: A client library for the Arduino Ethernet Shield that provides support for MQTT.



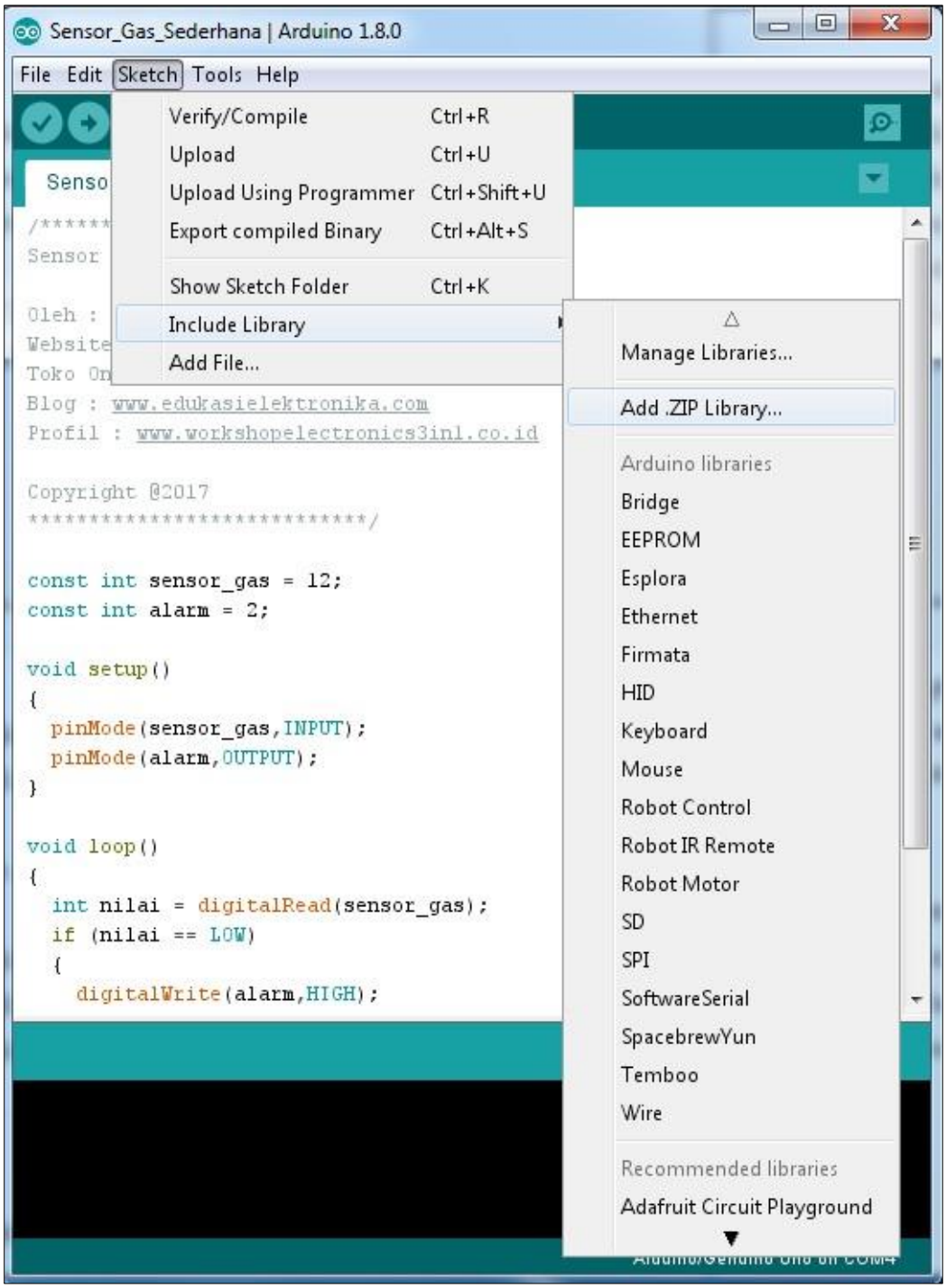
Gambar 3. 5 Download File Library PubSubClient

Lalu klik “Download ZIP”



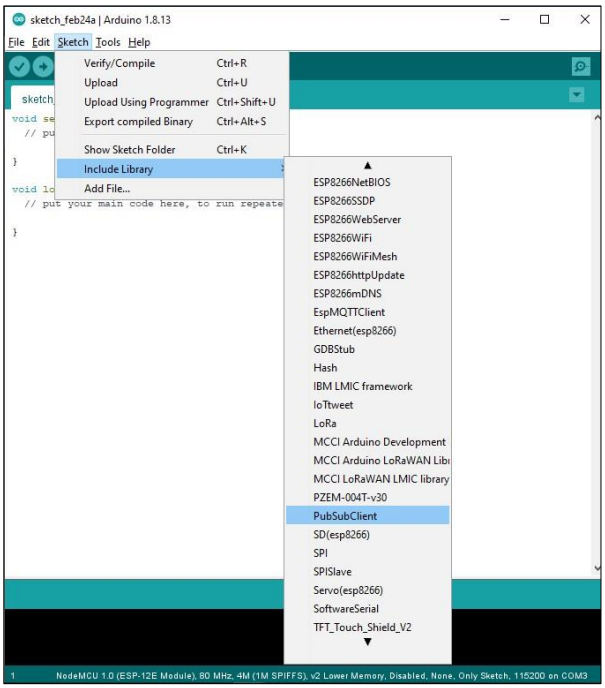
Gambar 3. 6 Klik *Download* ZIP

b. Buka Software atau Aplikasi **Arduino IDE.** Perhatikan gambar dibawah ini. Klik **add.ZIP** Library



Gambar 3. 7 Klik add.ZIP *Library*

c. Lalu pilih File Library **PubSubClient** yang sudah anda download tadi. Jika Berhasil akan muncul seperti gambar dibawah ini.



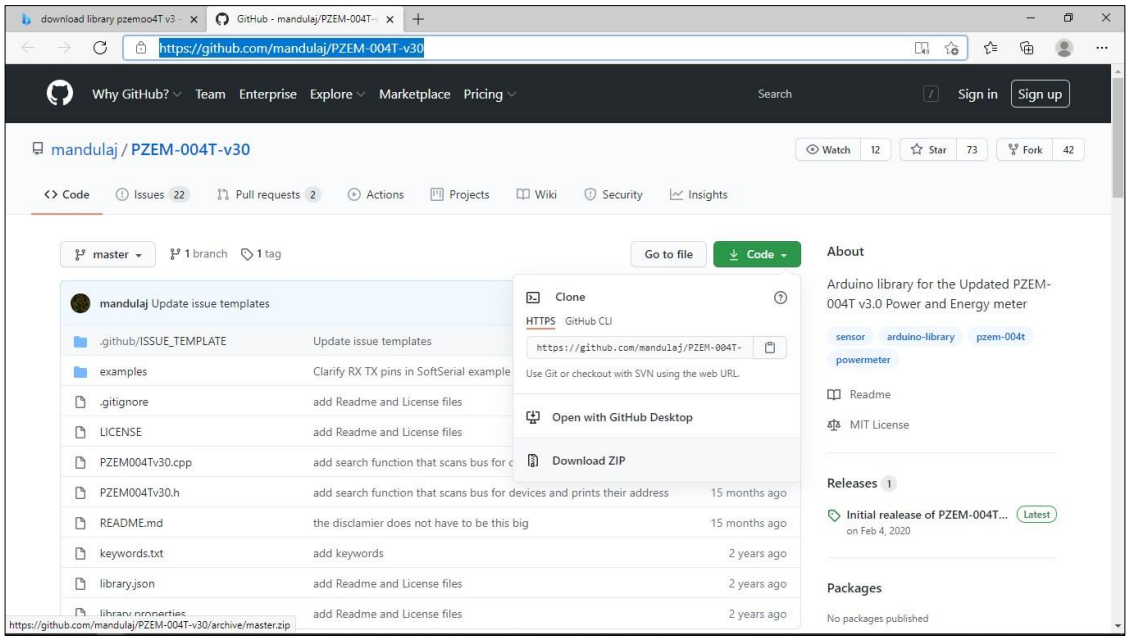
Gambar 3. 8 *File Library* PubSubClient sudah terpasang

d. Library siap digunakan.

3.3 Downlaod File Library PZEM004Tv30

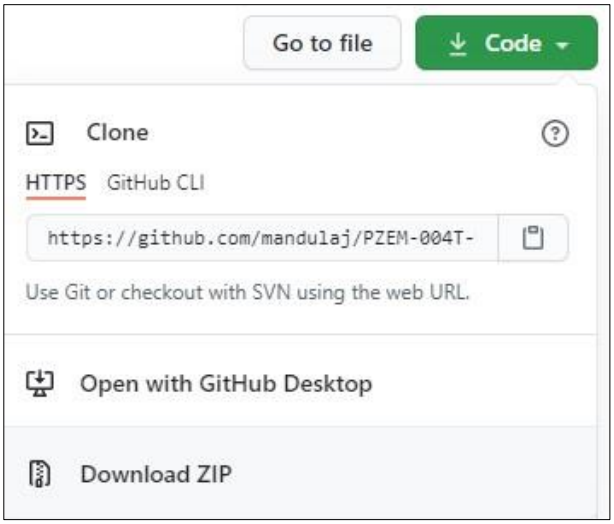
a. Download library PZEM004Tv30 dengan mengklik link di bawah :

GitHub - mandulaj/PZEM-004T-v30: Arduino library for the Updated PZEM-004T v3.0 Power and Energy meter



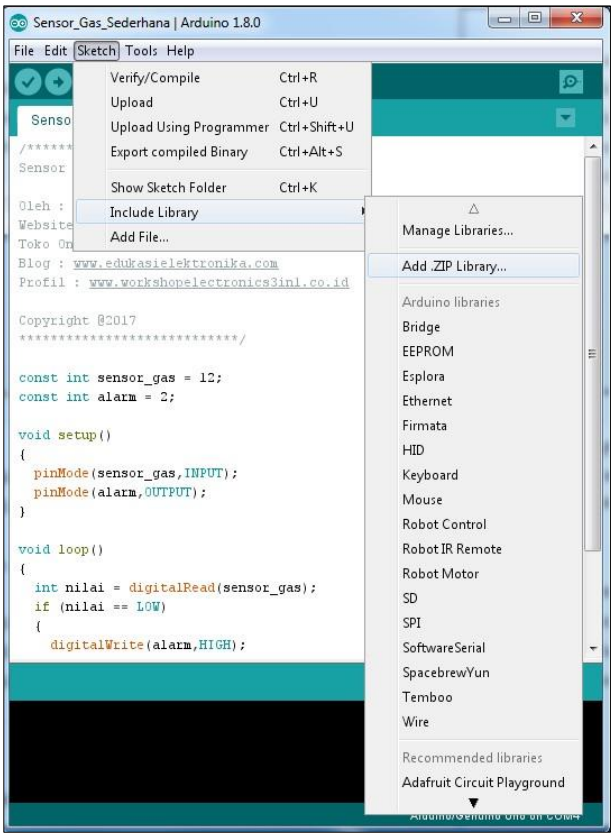
Gambar 3. 9 *Download File Library* PZEM004T

Lalu klik “Download ZIP”



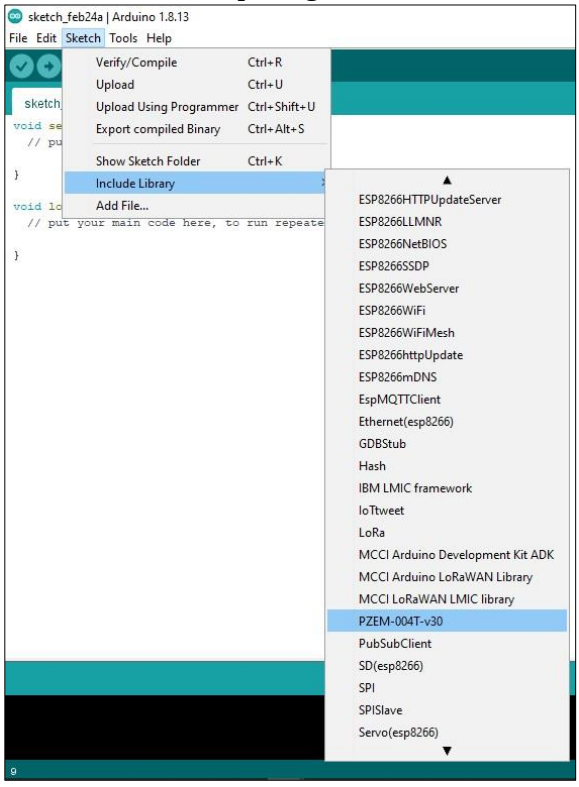
Gambar 3. 10 *Download* ZIP

b. Buka Software atau Aplikasi **Arduino IDE.** Perhatikan gambar dibawah ini Klik **add.ZIP** Library.



c. Lalu pilih File Library **PZEM004Tv30** yang sudah anda download tadi.

d. Jika Berhasil akan muncul seperti gambar dibawah ini.



e. Library siap digunakan.

BAB IV

CODING DAN PERAKITAN

4.1 Coding

1. Smart PJU
   a. Buka Arduino IDE
   b. Copy koding di bawah ini :

```
#include <PZEM004Tv30.h>
#include <PubSubClient.h>
#include <ESP8266WiFi.h>

const char* wifiSsid              = "LSKK_Lantai2";
const char* wifiPassword          = "lskk12345";
const char* mqttHost              = "rmq-pju.pptik.id";
const char* mqttUserName          = "/pju:pju"; const
char* mqttPassword          = "pEj3Uu123"; //const
char* mqttClient          = "Log";
const char* mqttQueuePJU          = "Log";

String deviceGuid     = "6626b0b5-a868-4f7e-9ebb-b2d68e124a35";
PZEM004Tv30 pzem(D2, D3); // Software Serial pin 11 (RX) & 12 (TX)
```

```
WiFiClient espClient;
PubSubClient client(espClient);
byte mac[6]; //array temp mac address
String MACAddress;
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
////

/*
 * Set up WiFi connection
 */
 void setup_wifi(){
  delay(10);
  //We start by connecting to a WiFi network
  Serial.println();
  Serial.print("Connecting to :");
  Serial.println(wifiSsid);
  WiFi.begin(wifiSsid, wifiPassword);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
```

```
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
 }
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
////

/*
 * Function for Get Mac Address from microcontroller
 */

String mac2String(byte ar[]) {
  String s;
  for (byte i = 0; i < 6; ++i)
  {
    char buf[3];
    sprintf(buf, "%2X", ar[i]);
    s += buf;
    if (i < 5) s += ':';
  }
  return s;
}
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
////

/*
 * Function for Print Mac Address
 */
 void printMACAddress() {
  WiFi.macAddress(mac);
  MACAddress = mac2String(mac);
  Serial.println(MACAddress);
}
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
////



/*
 * Function for Get message payload from MQTT rabbit mq
 */
void callback(char* topic, byte* payload, unsigned int length){   char
message[5]; //variable for temp payload message
  Serial.print("Message arrived in topic: ");
  Serial.println(topic);
Serial.print("Messagge :");
for(int i = 0;i < length;i++){
Serial.print((char)payload[i]);
```

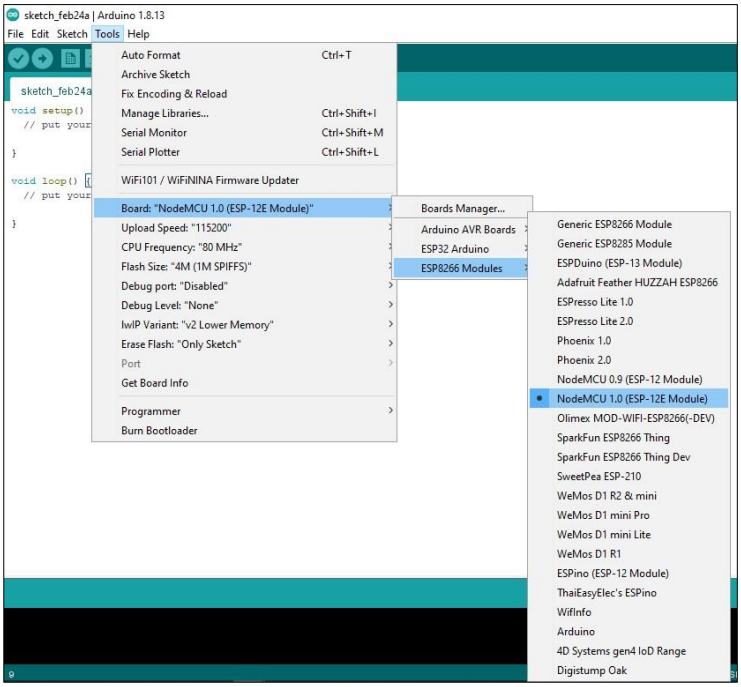
```
    message[i] = (char)payload[i]; //initiate value from payload to message variable
}

}
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
////

/*
 * Function for Reconnecting to MQTT/RabbitMQ
 */
void reconnect() {
  // Loop until we're reconnected
  printMACAddress();
  const char* CL;
  CL = MACAddress.c_str();
  Serial.println(CL);
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Attempt to connect
    if (client.connect(CL, mqttUserName, mqttPassword)) {
      Serial.println("connected");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      ESP.restart();
      delay(5000);

    }
  }
}

void setup() {
  Serial.begin(115200);
  setup_wifi();
  printMACAddress();
  client.setServer(mqttHost, 1883);
  client.setCallback(callback);
}

void loop() {
   if (!client.connected()) {
    reconnect();
  }

   float voltage = pzem.voltage();
    String convertvoltage = String(voltage);
    char dataToMQTTa[50];
    convertvoltage.toCharArray (dataToMQTTa, sizeof(dataToMQTTa));
   if(voltage != NAN){
```

```
    Serial.print("Voltage: ");
    Serial.print(voltage);
    Serial.println("V");
  } else {
    Serial.println("Error reading voltage");
  }

  float current = pzem.current();     String
convertcurrent = String(current);
   char dataToMQTTb[50];
   convertcurrent.toCharArray (dataToMQTTb, sizeof(dataToMQTTb));
if(current != NAN){       Serial.print("Current: ");
    Serial.print(current);
    Serial.println("A");
  } else {
    Serial.println("Error reading current");
  }
  float power = pzem.power();     String
convertpower = String(power);     char
dataToMQTTc[50];
   convertpower.toCharArray(dataToMQTTc, sizeof(dataToMQTTc));
if(current != NAN){       Serial.print("Power: ");
    Serial.print(power);
    Serial.println("W");
  } else {
    Serial.println("Error reading power");
  }


  float energy = pzem.energy();     String
convertenergy = String(energy);
   char dataToMQTTd[50];
   convertenergy.toCharArray        (dataToMQTTd,        sizeof(dataToMQTTd));
if(current != NAN){
    Serial.print("Energy: ");
    Serial.print(energy,3);
    Serial.println("kWh");
  } else {
    Serial.println("Error reading energy");
  }

  float frequency = pzem.frequency();     String
convertfrequency = String(frequency);
   char dataToMQTTe[50];
   convertfrequency.toCharArray (dataToMQTTe, sizeof(dataToMQTTe));
if(current != NAN){       Serial.print("Frequency: ");
    Serial.print(frequency, 1);
```

```
    Serial.println("Hz");
  } else {
    Serial.println("Error reading frequency");

String dataPJU = String(deviceGuid + "#" + voltage + "#" + current + "#" + power
  "#" + energy + "#" + frequency);
client.publish(mqttQueuePJU, dataPJU.c_str());
client.loop();
delay(5100);

}

}
```

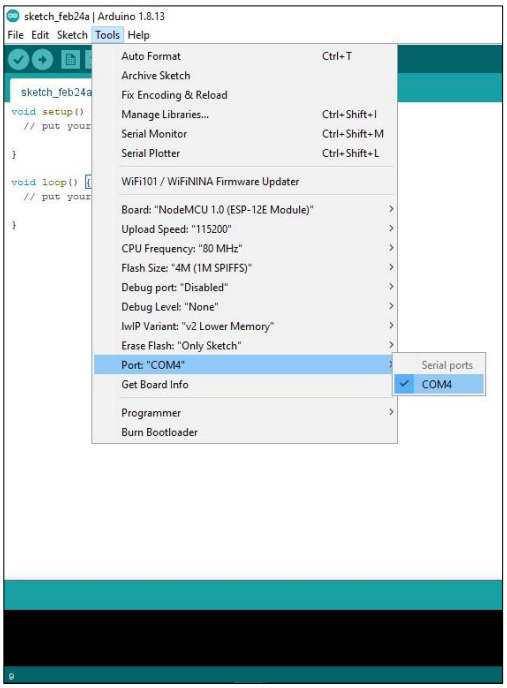
c. Paste ke Arduino IDE laptop/PC anda.

d. Sesuaikan board pada Arduino IDE dengan board yang anda gunakan



e. Hubungkan NodeMCU ESP8266 ke laptop/PC menggunakan kabel USB

f. Atur port pada Arduino IDE lalu sesuiakan dengan port yang anda gunakan



g. Klik upload pada Arduino IDE di pojok kiri atas.

h. Jika sudah selesai upload coding, lepas kabel USB dari NodeMCU

2. Aktuator

a. Buka Arduino IDE

b. Copy koding di bawah ini :

```
/*
   Develop by     : Fahmi Nurfadilah
   Email          : fahmi.nurfadilah1412@gmail.com
   Updated by     : Ary Setijadi Prihatmanto
   Email          : asetijadi@lskk.ee.itb.ac.id
   Project        : LSKK Homeautomation
   Version        : 2.1
*/

#include <PubSubClient.h>
#include <ESP8266WiFi.h>

// Update these with values suitable for your network.
```

```
const char* ssid = "LSKKHomeauto";
const char* password = "1234567890";
const char* mqtt_server = "rmq-pju.pptik.id";
const char* mqtt_user = "/pju/pju";
const char* mqtt_pass = "p3JeUu123";
const char* mqtt_pub_topic = "Aktuator";
String statusDevice = "0";
int relay1 = D1 ;



const char* device_guid = 5b0ed6a6-8e02-11eb-8dcd-0242ac130003 ";
String output_value;

char msg[100];
WiFiClient espClient;
PubSubClient client(espClient);

byte mac[6];
String MACAddress;

void setup_wifi() {
  delay(10);
  // We start by connecting to a WiFi network
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
```

```
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}

String mac2String(byte ar[]) {
  String s;
  for (byte i = 0; i < 6; ++i)
  {
    char buf[3];
    sprintf(buf, "%2X", ar[i]);
    s += buf;
    if (i < 5) s += ':';
  }
  return s;
}

void printMACAddress() {
  WiFi.macAddress(mac);
  MACAddress = mac2String(mac);
  Serial.println(MACAddress);
}
char sPayload[100];
char message [40] ;
char address[40];
void callback(char* topic, byte* payload, unsigned int length) {
  memcpy(sPayload,payload,length);
  memcpy(address,payload,36);
  memcpy(message,&payload[37],length-37);
  Serial.print("Message arrived [");
  Serial.print(sPayload);
  Serial.println("] ");
```

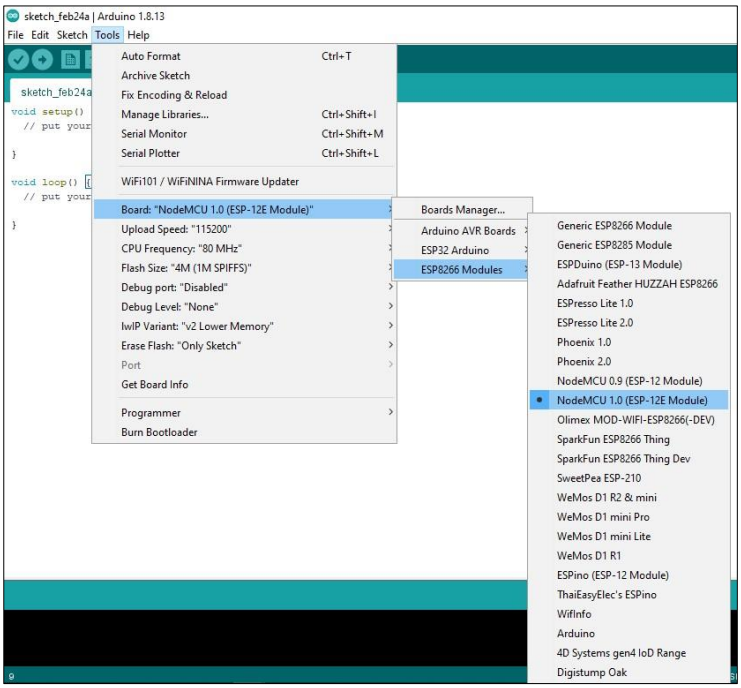
```
Serial.println(device_guid);
Serial.println(address);
if(String((char *)address) == String((char *)device_guid))
{
  Serial.println("tersambung");
}
else
{
  Serial.println("tidak tersambung");
  return;
}

Serial.println(message);

if (message[0] == '1') {
  digitalWrite(relay1, LOW);
  Serial.println("relay 1 on");
  statusDevice = "1";

}
if (message[0] == '0') {
  digitalWrite(relay1, HIGH);
  Serial.println("relay 1 off");
  statusDevice = "0";
}




Serial.print("Publish message: ");
String pubmsg = String(device_guid) + "#" + String(statusDevice);
Serial.println(pubmsg);
client.publish(mqtt_pub_topic, pubmsg.c_str());
delay(50);
```

```
}

void reconnect() {
  // Loop until we're reconnected
  printMACAddress();
  const char* CL;
  CL = MACAddress.c_str();
  Serial.println(CL);
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Attempt to connect
    if (client.connect(CL, mqtt_user, mqtt_pass)) {
      Serial.println("connected");
      client.subscribe("Aktuator"); //mengambil data dari antrian received
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      ESP.restart();
      delay(5000);

    }
  }
}

void setup()
{
  pinMode(D1, OUTPUT);
  digitalWrite(D1, HIGH);
  //pinMode(input, INPUT);
  Serial.begin(115200);
  setup_wifi();
  printMACAddress();
  client.setServer(mqtt_server, 1883);
```

```
  client.setCallback(callback);
 }
 void loop() {
  if (!client.connected()) {
   reconnect();
  }
  client.loop();
 }
```

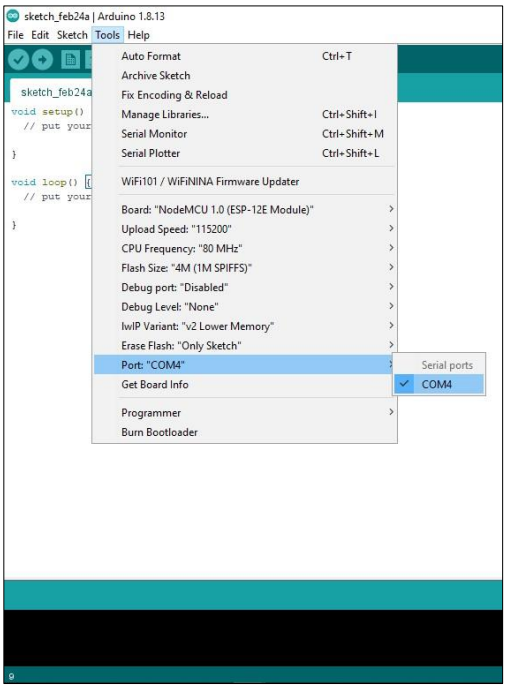
c. Paste ke Arduino IDE laptop/PC anda.

d. Sesuaikan board pada Arduino IDE dengan board yang anda gunakan



e. Hubungkan NodeMCU ESP8266 ke laptop/PC menggunakan kabel USB

f. Atur port pada Arduino IDE lalu sesuiakan dengan port yang anda gunakan



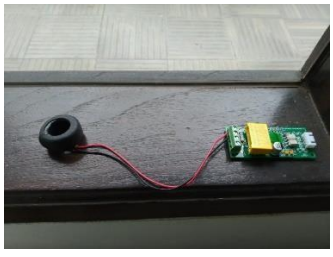
g. Klik upload pada Arduino IDE di pojok kiri atas.

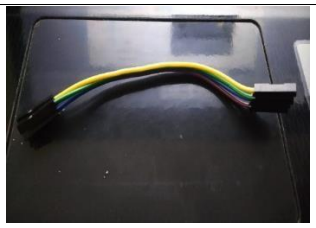
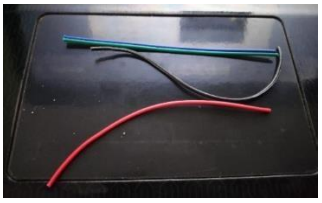
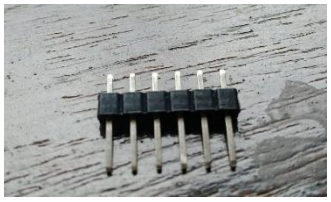
Jika sudah selesai upload coding, lepas kabel USB dari NodeMCU

4.2 Perakitan

4.2.1 Alat dan Bahan

Alat dan Bahan yang digunakan adalah :

Nama Komponen	Kuantitas	Gambar Komponen
NodeMCU ESP8266 V.3	1	
PZEM-004T V.3 100A	1	

Modul relay	1	
Jumper femalefemale	4	
Jumper	4	
Pin *header female*	15x2	
Pin *header male*	4	
Power supply DC 5V 3A	1	
Kabel serabut	10 meter	

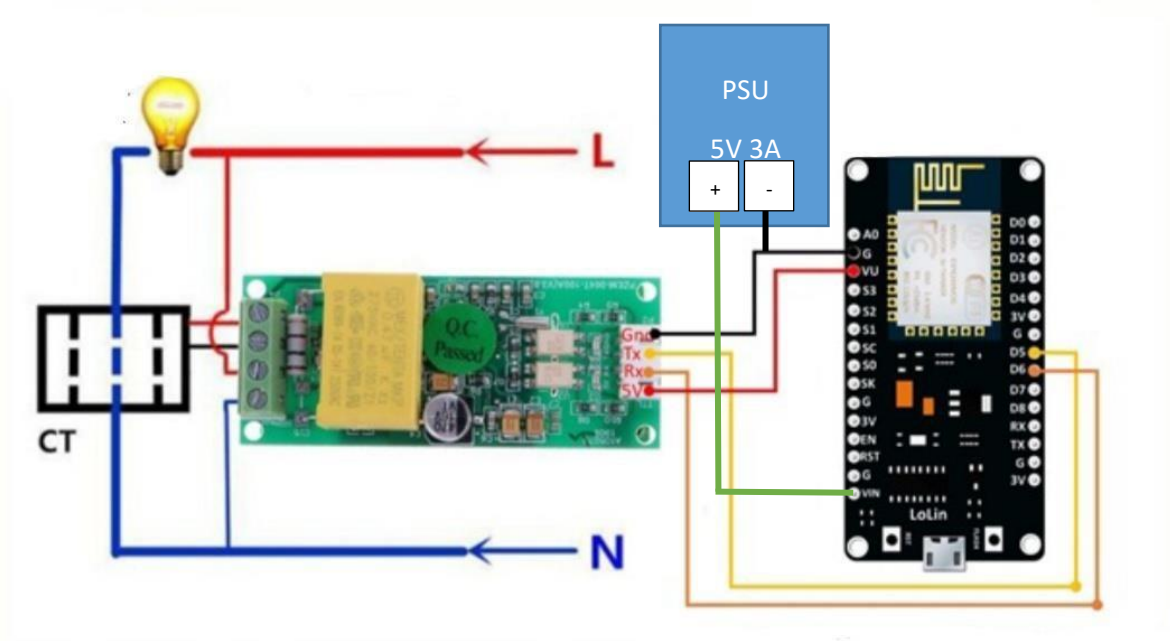
PCB bolong	1	
Timah	1	
Pasta solder	1	
Pemaketan *box* hitam	2	

Nama Alat	**Kuantitas**	**Gambar Alat**
Solder	1	
Sabut besi	1	
Pemotong	1	

Bor duduk	1	
Cutter	1	

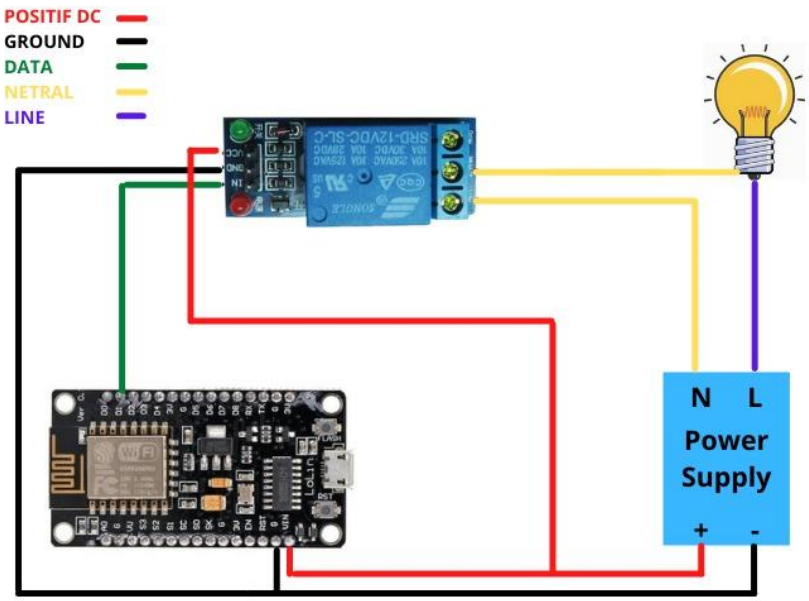
4.2.2 Langkah Perakitan

a. Skema rangkaian Smart PJU



b. Buatlah rangkaian seperti pada gambar skema rangkaian dengan menyiapkan alat dan bahan terlebih dahulu

c. Tambahkan rangkaian aktuator.



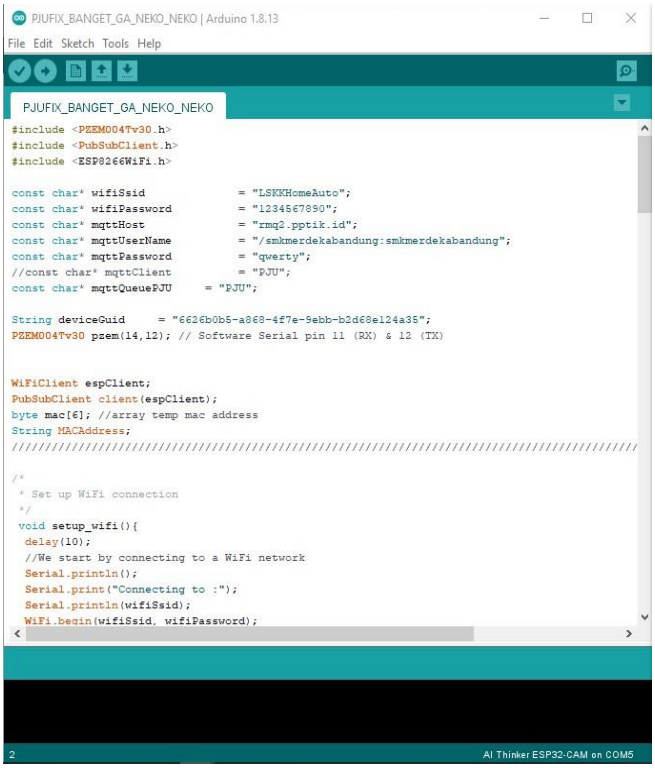
BAB V

TESTING

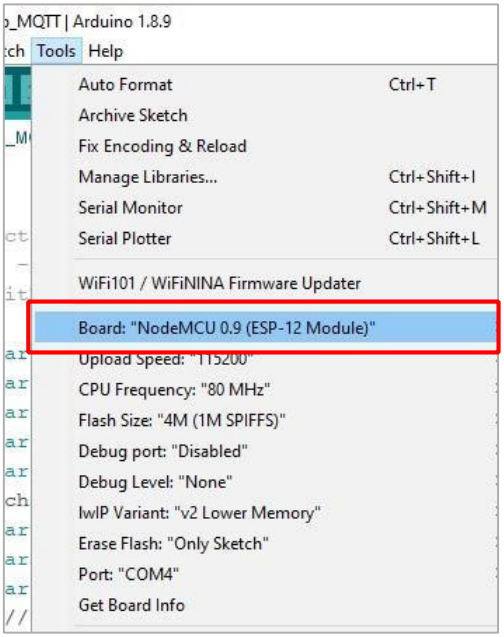
5.1 Serial Monitor

Berikut ini merupakan langkah – langkah pengujian Smart PJU pada serial monitor Arduino IDE.

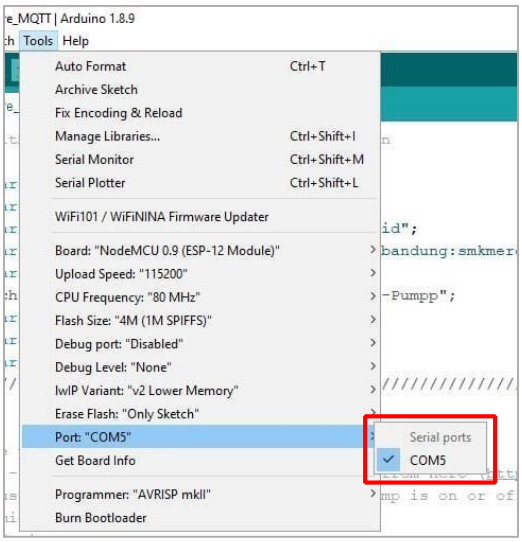
a. Siapkan laptop serta unduh dan pasang aplikasi Arduino IDE pada laptop.
b. Buka 2 *skecth* Arduino IDE, satu untuk kode program Aktuator dan *sketch* lainnya untuk kode program Smart PJU atau pompa.
c. Masukkan sketch atau kode program Smart PJU dan Aktuator



d. Klik Menu *Tools* → pilih board NodeMCU 0.9 (ESP-12 Module) pada masing – masing *sketch*.



e. Upload masing – masing kode program pada board NodeMCU dan pastikan terhubung pada *serial port* masing – masing agar dapat melihat hasil monitoring melalui *serial monitor*.



f. Cek serial monitor dengan mengklik icon serial monitor dan data akan tampil pada serial monitar lalu reset board nodeMCU dan Wemos.

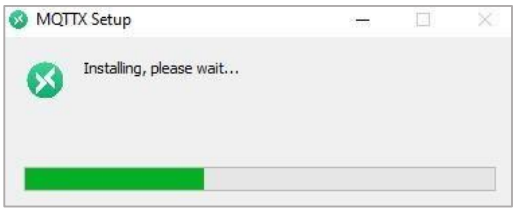
4.4.2 Pengujian menggunakan MQTTX

MQTTX merupakan sebuah plaform yang digunakan untuk menguji konektivitas MQTT. Berikut langkah – langkah serta hasil pengujian menggunakan media MQTTX :

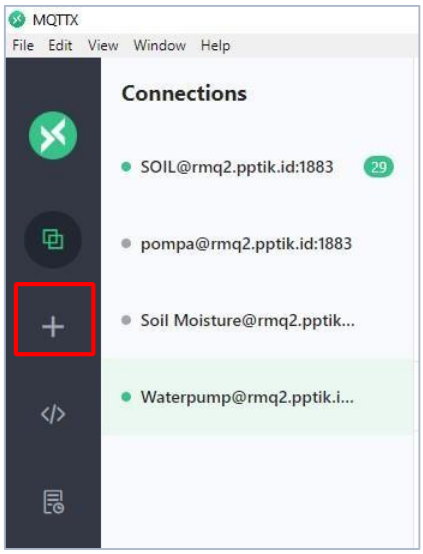
1. Langkah Pengujian Aktuator Smart PJU

Berikut ini merupakan langkah – langkah pengujian Aktuator Smart PJU pada MQTTX :

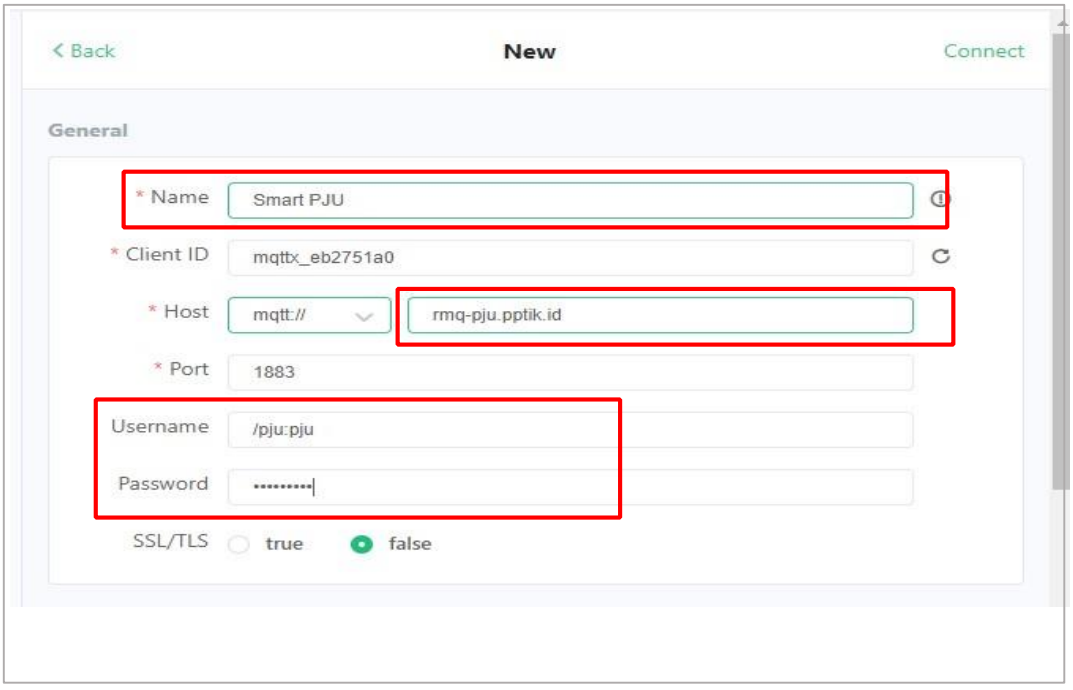
a. Unduh dan pasang MQTTX pada laptop.



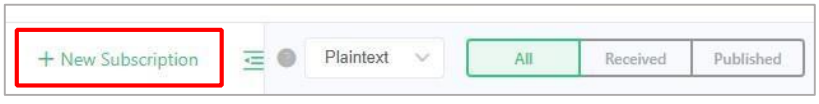
b. Buka MQTTX. Pada MQTTX klik *icon* menu + untuk menambah *client* atau sambungan pada MQTTX.



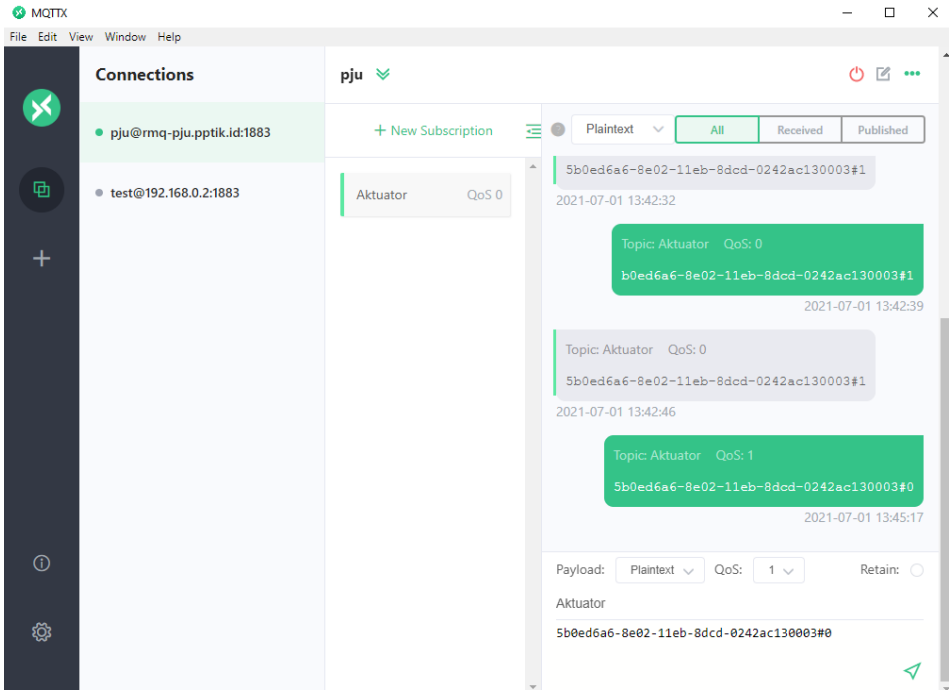
c. Input data pada MQTT dengan mengikuti kode program yang telah dibuat pada Arduino IDE seperti gambar berikut.



e. Pada MQTTX Klik *New Subscription* → buat topik → klik *confirm*.



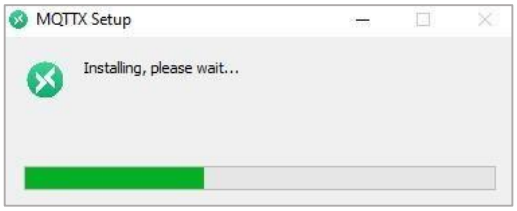
f. Lalu akan tampil datanya. Untuk menyalakan actuator dapat publish "deviceguid#1" dan untuk mematikannya "deviceguid#0" seperti gambar berikut



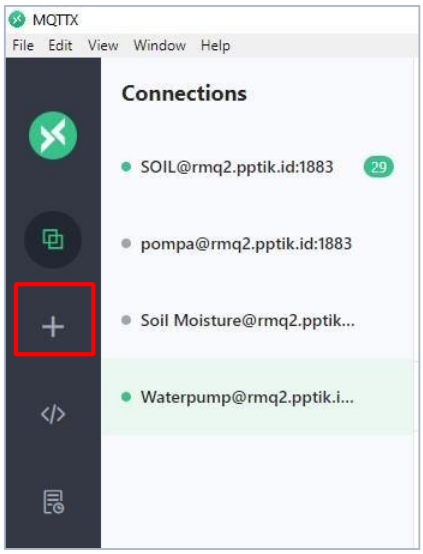
2. Langkah Pengujian Smart PJU

Berikut ini merupakan langkah – langkah pengujian *smart* PJU pada MQTTX :

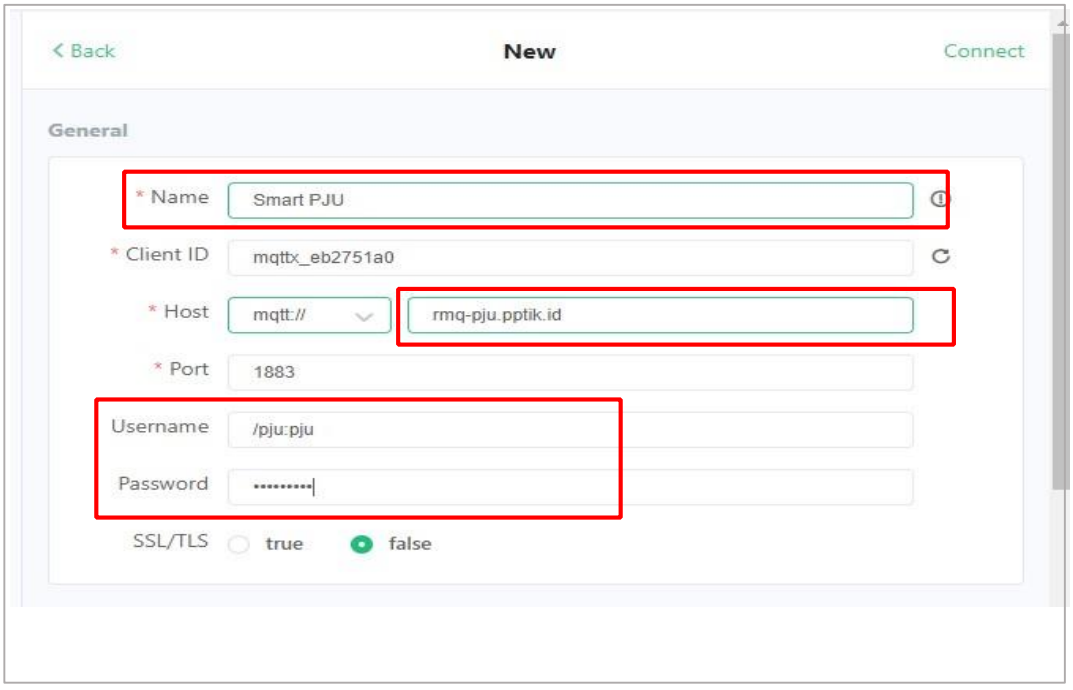
a. Unduh dan pasang MQTTX pada laptop.



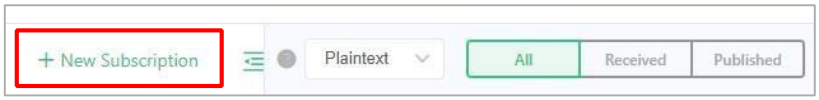
b. Buka MQTTX. Pada MQTTX klik *icon* menu + untuk menambah *client* atau sambungan pada MQTTX.



c. Input data pada MQTT dengan mengikuti kode program yang telah dibuat pada Arduino IDE seperti gambar berikut.

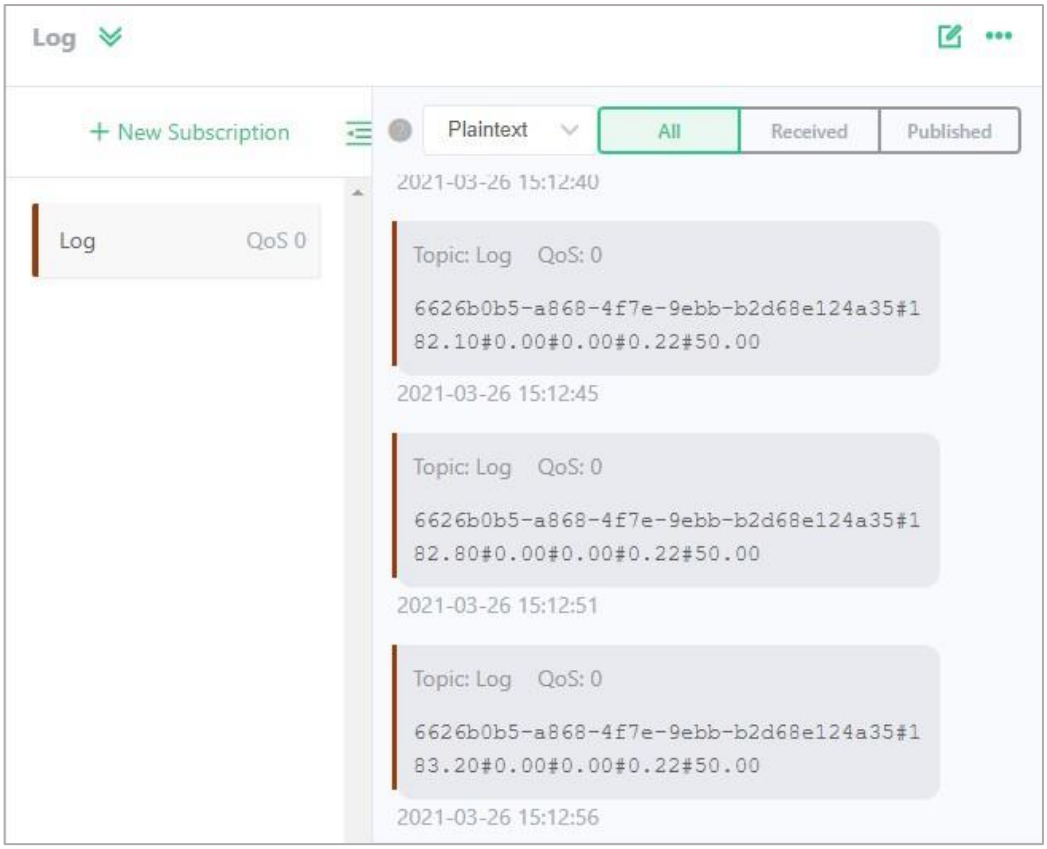


g. Pada MQTTX Klik *New Subscription* → buat topik → klik *confirm*.



h. Pada MQTTX data akan tampil setelah membuat topik lalu klik confirm. Sedangkan pada MQTTBox data *publish* serta data *subscibe* baik dari sensor,

aktuator atau secara manual akan tampil pada MQTTBox setelah mengklik *subscribe*.



4.4.3 Pengujian pada web RabbitMQ

RabbitMQ merupakan salah satu *message broker* yang digunakan pada protokol MQTT.

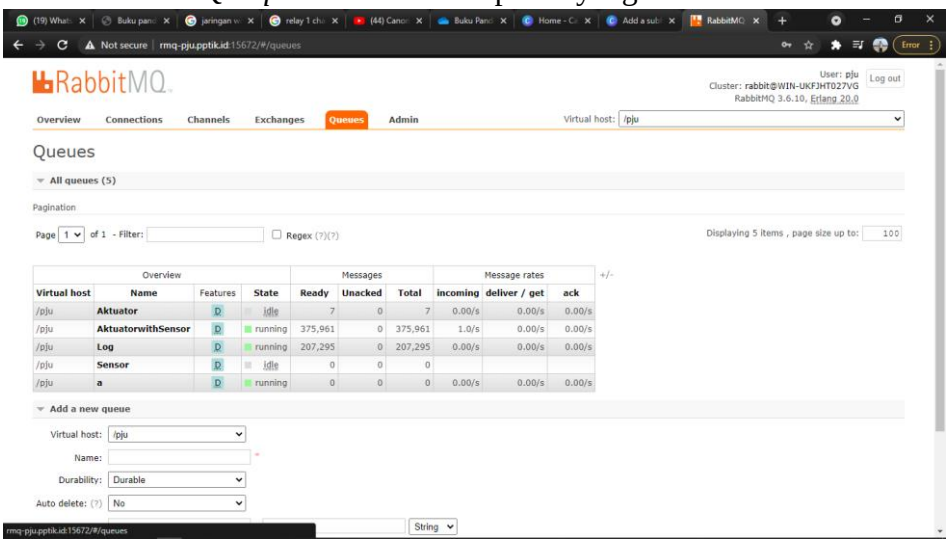
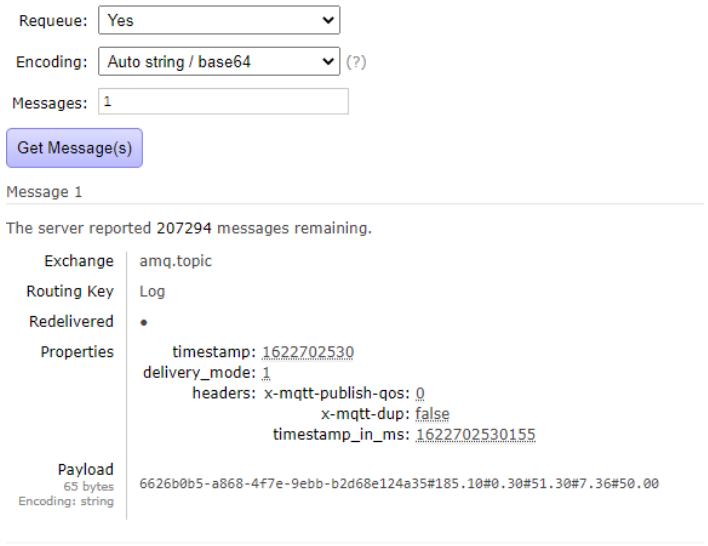
1. Langkah Pengujian

Berikut ini merupakan langkah – langkah pengujian Smart PJU pada web rabbitMQ :

a. Masuk pada alamat server *website* RabbitMQ yang telah disediakan oleh PPTIK – ITB pada linnk http://rmq-pju.pptik.id:15672
b. Login *username* yaitu “pju” dan paswordnya “p3JeUu123”.

c. Masukkan *virtual host* : /pju.

d. Klik menu *Queque* untuk melihat pesan yang diterima.



e. Klik topik atau nama queque log lalu scroll ke bawah dan klik Get Message(s) untuk melihat hasil dari data PJU